Date=26/08/2020 Lecture By=Shubham Joshi Subject ⇒ DP=4

IN PREVIOUS LECTURE (QUICK RECAP) Date-25/08/2020	In Today's Lecture (Overview)
Question=1 Find the longest Increasing subsequence	Kadane's Algorithm:
Question=2 Find the Longest Increasing SubArray	Question=1⇒ Largest Sum Contiguous Subarray
MCQs Questions for Self Practice // CC for the Day	Write an efficient program to find the sum of contiguous subarray within a one-dimensional array of numbers which has the largest sum
	<u>Mcqs</u>
	Questions For self Practice // Assignment For the Day

In today's lecture we discussed A very Important Algorithm Called Kadane's Algorithm And we also solved one problem using the kadane's Algorithm So In Today's notes Kadane's Algorithm And problem that was solved In the lecture will Be Discussed

Kadane's Algorithm:

```
max_ending_here = 0
return max_so_far
```

#This Is just the example of How Kadane's Algorithm Works **Explanation:**

Simple idea of Kadane's algorithm is to look for all positive contiguous segments of the array (max_ending_here is used for this). And keep track of the maximum sum contiguous segment among all positive segments (max_so_far is used for this). Each time we get a positive sum compare it with max_so_far and update max_so_far if it is greater than max_so_far

Lets Learn this Algorithm By Simple Example

```
Let's take the example:

{-2, -3, 4, -1, -2, 1, 5, -3}

max_so_far = max_ending_here = 0

for i=0, a[0] = -2

max_ending_here = max_ending_here + (-2)

Set max_ending_here = 0 because max_ending_here < 0

for i=1, a[1] = -3

max_ending_here = max_ending_here + (-3)

Set max_ending_here = 0 because max_ending_here < 0

for i=2, a[2] = 4

max_ending_here = max_ending_here + (4)

max_ending_here = 4

max_so_far is updated to 4 because max_ending_here greater than max_so_far which was 0 till now

for i=3, a[3] = -1

max_ending_here = max_ending_here + (-1)

max_ending_here = 3
```

```
for i=4, a[4] = -2
max_ending_here = max_ending_here + (-2)
max_ending_here = 1

for i=5, a[5] = 1
max_ending_here = max_ending_here + (1)
max_ending_here = 2

for i=6, a[6] = 5
max_ending_here = max_ending_here + (5)
max_ending_here = 7
max_so_far is updated to 7 because max_ending_here is
greater than max_so_far

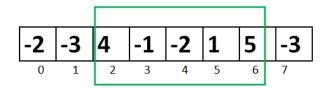
for i=7, a[7] = -3
max_ending_here = max_ending_here + (-3)
max_ending_here = 4
```

#This is the Example of How the Kadane's Works Now Lets See the Problem Regarding the Kadane's Algorithm

Question=1⇒ Largest Sum Contiguous Subarray

Write an efficient program to find the sum of contiguous subarray within a one-dimensional array of numbers which has the largest sum

Largest Subarray Sum Problem



$$4 + (-1) + (-2) + 1 + 5 = 7$$

Maximum Contiguous Array Sum is 7

Code For the Program

```
# Fython program to find maximum contiguous subarray
# Function to find the maximum contiguous subarray
from sys import maxint

def maxSubArraySum(a, size):

    max_so_far = -maxint - 1
    max_ending_here = 0

for i in range(0, size):
    max_ending_here = max_ending_here + a[i]
    if (max_so_far < max_ending_here):
        max_so_far = max_ending_here

    if max_ending_here < 0:
        max_ending_here = 0

    return max_so_far

# Driver function to check the above function
a = [-13, -3, -25, -20, -3, -16, -23, -12, -5, -22, -15, -4, -7]
print ("Maximum contiguous sum is", maxSubArraySum(a,len(a))</pre>
```

This code Is from Geeksforgeeks

https://www.geeksforgeeks.org/largest-sum-contiguous-subarray/

Mcqs

1. What is the time complexity of kadane algo?

```
A=O(n2)
B=O(n)
C=O(logn)
```

2.what determines the space complexity of dp?
A=function calls
B=state of the dp
3. What is the space time complexity of kadane algo?
A=O(nlogn)
B=O(1)
C=O(n)
4.which of these is not a subsequence of array [12,34,55,66]
A=[55,66]
B=[34,66]
C=[55,34]

Questions For self Practice // Assignment For the Day

https://leetcode.com/problems/minimum-path-sum/

https://leetcode.com/problems/decode-ways/